Adaptively Detecting Changes in Autonomic Grid Computing

Xiangliang Zhang, Cecile Germain-Renaud, Michèle Sebag

▶ **To cite this version:**

Xiangliang Zhang, Cecile Germain-Renaud, Michèle Sebag. Adaptively Detecting Changes in Autonomic Grid Computing. Procs of ACS 2010, Oct 2010, Belgium. http://wiki.esi.ac.uk/ACS2010. hal-00540579

HAL Id: hal-00540579
https://hal.science/hal-00540579

Submitted on 27 Nov 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Adaptively Detecting Changes in Autonomic Grid Computing

Xiangliang Zhang*
Mathematical and Computer Sciences and Engineering Division
King Abdullah University of Science and Technology, Saudi Arabia
xiangliangzhang@gmail.com

Cecile Germain Michele Sebag
TAO - LRI, CNRS, INRIA
Université Paris-Sud 11, France
{cecile,sebag}@lri.fr

***Abstract*—Detecting the changes is the common issue in many application fields due to the non-stationary distribution of the applicative data, e.g., sensor network signals, web logs and grid-running logs. Toward Autonomic Grid Computing, adaptively detecting the changes in a grid system can help to alarm the anomalies, clean the noises, and report the new patterns. In this paper, we proposed an approach of self-adaptive change detection based on the Page-Hinkley statistic test. It handles the non-stationary distribution without the assumption of data distribution and the empirical setting of parameters. We validate the approach on the EGEE streaming jobs, and report its better performance on achieving higher accuracy comparing to the other change detection methods. Meanwhile this change detection process could help to discover the device fault which was not claimed in the system logs.**

I. Introduction

Autonomic Computing (AC) is acknowledged as a key topic for the economy of computational systems, in terms of both power consumption and resource allocation [1], and human labor and maintenance support [2]. To realize these autonomic functionalities, the system behavior is analyzed to provide the principles for decision-making. For example, in EGEE grid, we exploited the gLite reports on the lifecycle of the jobs and on the behavior of the middleware components for providing the summarized information of grid running status [3].

One special issue in the analysis of system behavior is the **non-stationary distribution**. In EGEE grid, for example, on each day more than 300,000 jobs are submitted by different users from various fields. The behaviors of jobs are in non-stationary distribution due to the evolution of the phenomenon, e.g., the traffic, the users and the modes of usage. More generally, the non-stationary distribution exists in many other domains, such as sensor network signals, web logs and computer network traffic.

One challenge of handling the data in non-stationary distribution is to detect the changes in the underlying data distribution. The detection of changes can help for i) anomaly detection – triggers alerts/alarms; ii) data cleaning – detects errors in data feeds; iii) data mining – indicates when to learn a new model.

The general idea for detecting "changes" is to compare a reference distribution with a current window of events. To detect item changes, for example, individual items with big frequency change, sketches based techniques can be used. Another widely used approach is non-parametric change detection. It has few parameters to set, but must specify when to call a change significant as did in [4], [5].

The approach proposed in [6] detects changes by analyzing the trend of data. Data density estimated by *Kernel density estimation* (KDE) and changes of data density estimated by *Velocity density estimation* (VDE) are used to discover the evolution of data: *dissolution*, *coagulation* and *shift*.

In [4], Dasu et al. use relative entropy, also called the Kullback-Leibler distance, to measure the difference between two given distributions. The first step of this approach is to partition the reference data (a part of the firstly arrived data), and compute the discrete probability p over the partitioned regions. Then over a window of recent streaming items, the same space division is applied and discrete probability q is computed. The KL divergence of recent stream items and the reference data is computed by $D(p||q) = \sum_x p(x)log_2p(x)/q(x)$. A statistical inference procedure based on the theory of bootstrapping is used to tell the significant of KL divergence.

In [5], a statistical test called the *density test* is defined for detecting changes, saying whether the newly observed data points S' are sampled from the underlying distribution that produced the baseline data set S. This test statistic is strictly distribution-free under the null hypothesis.

All the above methods of change detection are based on stationary reference data. In many application areas, however, the data is typically streaming and not stationary. For example, in network intrusion detection by monitoring network traffic, the distribution of reference data (usually normal data) is evolving over time.

In this paper, we propose a self-adaptive method of change detection based on the analysis of *outliers* that were discovered by a dynamic online models. We use our previously proposed algorithm StrAP [7] for online discovering the *outliers* as StrAP gives better performance in terms of accuracy than other online clustering methods. The *outliers* are the arrived data items which deviate from the reference model. The appearance of *outliers* can be caused by the normal concept drift, or the abnormal behaviors or the noise. In this paper, we analyze the outliers and detect the changes in the non-stationary data distribution with the purpose of identifying

*Most of this work was completed when the author was a PhD student at University Paris Sud 11, France.

the possible reasons causing the changes. Coupling with the Autonomic Computing, the proposed method adaptively detects the changes based on the online updating model and the self-adaption of the algorithm parameters according to the data distribution.

The rest of this paper is organized as follows. In Section II, we describe our proposed self-adapative approach for change detection. Section III shows the experimental results applied on EGEE jobs. Finally, we conclude and give our perspectives in Section IV.

II. Self-adaptive Change Detection Algorithm

Our proposed approach is based on a statistical change point detection test, the so-called Page-Hinkley test (PH) [8], [9]. We firstly introduce this PH statistical test method.

A. Page-Hinkley statistical test

Formally, let p_t denote the observation of the variable in non-stationary distribution. To detect the changes of variable p_t, the PH test is controlled after a detection threshold λ and tolerance δ, as follows:

$$\begin{aligned} \bar{p}_t &= \frac{1}{t}\sum_{\ell=1}^{t} p_\ell \\ m_t &= \sum_{\ell=1}^{t} (p_\ell - \bar{p}_\ell + \delta) \\ M_t &= max\{m_\ell, \ell = 1...t\} \\ PH_t &= M_t - m_t \\ \texttt{If} \quad PH_t &> \lambda, \quad \text{change is detected} \end{aligned} \tag{1}$$

We give an example for showing how PH is used to detect the changes in Fig. 1. In this figure, red line p_t is the non-stationary distribution (change happened after 300). $\bar{p}_t$, m_t and M_t are computed from equation (1), where δ is usually set to a very small value, e.g.,10^{-2}. The gap between M_t and m_t, i.e. PH_t, keeps increasing after the change happened at 300. A threshold λ can be set to report the detected change.

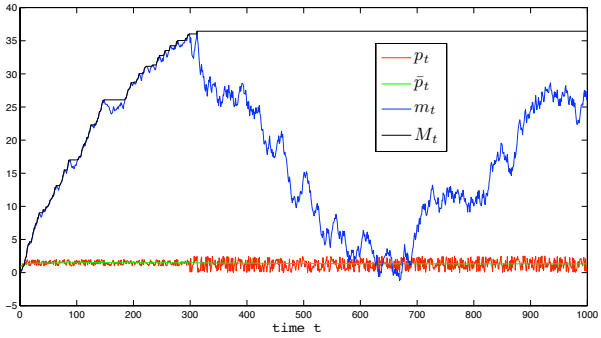

Fig. 1. The demonstration of change detection by PH

The threshold λ is an important parameter for detecting the changes in non-stationary distribution. Fixing λ by an empirical value is unfavorable for the detection of changes. A too large value of λ would delay or miss the detection of changes, while a too small one would falsely alarm the changes frequently. A self-adpated threshold λ according to the evolving distribution catches better the changes.

In [10], we have considered the λ adaption problem as a sequential control task that can be handled using an exploration-exploitation algorithm. The goal is to adjust the setting of λ to obtain better clustering model. Therefore, we define the observation object p_t as the proportion of outliers comparing with the clustering model. The method consists of recording a quality measurement of the clustering obtained and scoring each single value of λ that have been used. The quality of clustering model is measured by the Bayesian Information Criterion (BIC) [11], which is defined as the *distortion loss of the model* + the *size of model* + the *proportion of outliers*. In order to find out the appropriate setting of λ for minimizing the BIC cost, we used two approaches. One is e-greedy action selection approach for selecting discrete value for λ. The other is Gaussian Process regression approach for generating continuous value for λ.

This previous approach of self-adaptive change detection formalizes the adaptation of the threshold λ by an optimization problem. However, there are several difficulties. The first difficulty is that it defines the observation object p_t as the proportion of outliers w.r.t. the clustering model. This definition of p_t ignores the time-dependence of outliers, which is important for categorizing outliers. The densely arriving outliers would be the *normal concept drift* while irregularly arriving outliers can be *noise* or *anomalies*. The second difficulty is that the BIC object function for optimization has to be computed frequently, increasing the computational cost.

We will adapt the threshold λ according to the definition of observation object p_t.

B. Definition of p_t based on weighted standard deviation

Let us consider the sequence of outliers u_t in a non-stationary distribution. If u_t comes densely with similar values, they could be a new pattern. If u_t irregularly comes with values in a large range, they can be noise or rare anomalies. Therefore, we define the weighted standard deviation τ_t by simultaneously considering the value of u_t and the time l_t when u_t appears.

$$\tau_t = \sqrt{\frac{1}{t}\sum_{i=1}^{t} \omega_i (u_i - \bar{u})^2}$$

where $\bar{u} = \frac{1}{t}\sum_{i=1}^{t} u_i$ is the mean of u_i, and the coefficient $\omega_i = log(l_i - l_{i-1}) + 1$, l_i and l_{i-1} are the time when u_i and u_{i-1} come. In a special case when $\omega_i \equiv 1$ or $l_i - l_{i-1} = 1$ (u_i comes uniformly), weighted standard deviation (std) is the normal definition of std.

When u_t comes densely with similar values, τ_t will keep decreasing towards 0. PH can be used to detect the changing trend of τ_t, by defining p_t as the weighted std τ_t computed in a sliding window along u_t.

C. Self-adaptive threshold λ

Using PH for detecting the changes of u_t, the change reporting threshold λ is expected to be adapted in real-time. In

other words, this threshold λ_t is computed at each time step t.

From equation (1), we know that PH_t is a non-negative variable. The perfect case of setting threshold λ is to adapt it according to the trend of new-arriving items.

Proposition 1: The threshold for detecting changes by PH can be computed as

$$\lambda_t = \begin{cases} 0 & \text{if } PH_t = 0 \\ f * \bar{p}_t & \text{otherwise} \end{cases}$$

or

$$\lambda_{t_0} = \begin{cases} 0 & \text{if } PH_t = 0 \\ f * \bar{p}_{t_0} & \text{otherwise} \end{cases}$$

where, f is a constant called the λ factor, which is the number of required witnesses seeing the changes, e.g., $f = 30$. t_0 is the moment since when $PH_{t_0} \neq 0$. $\bar{p}_t$ and $\bar{p}_{t_0}$ are the moving average computed after equation (1).

Proof: The detection of change is triggered when $PH_t > \lambda$. In order to see how to set λ, firstly we have a look at how the non-negative PH_t increases. As defined in equation (1), if $PH_{t-1}, PH_t \neq 0$, the increase from PH_{t-1} up to PH_t is

$$PH_t - PH_{t-1} = (M_t - m_t) - (M_{t-1} - m_{t-1})$$

because $PH_{t-1}, PH_t > 0$, we have $M_t \equiv M_{t-1}$. Then

$$PH_t - PH_{t-1} = m_{t-1} - m_t = m_{t-1} - (m_{t-1} + p_t - \bar{p}_t + \delta)$$
$$= \bar{p}_t - p_t - \delta$$

Therefore, since $PH_{t_0} > 0$ (i.e., $\bar{p}_{t_0} > p_{t_0} + \delta$), for $t \geq t_0$, we have

$$PH_t = \sum_{i=t_0}^{t} (\bar{p}_i - p_i - \delta) \tag{2}$$

From equation (2), we can see that PH_t is the collection of deviation of p_t from $\bar{p}_t$. The scenario of changes happening is the weighted std p_t decreasing towards 0. To be sure of the effective changes, not fake anomalies, evidence should be collected in longer time. We define a λ factor, called f, to be the number of steps when PH_t keeps increasing. As p_t is decreasing towards 0, δ is a very small value (10^{-2}), and $\bar{p}_t$ decreases slowly, after f steps, $PH_t \approx f * \bar{p}_t$. Therefore, the first option for setting λ is

$$\lambda_t = f * \bar{p}_t$$

To avoid computing λ_t frequently, it can be set immediately when $PH_t > 0$. Then the second way for setting λ is

$$\lambda_{t_0} = f * \bar{p}_{t_0}$$

where t_0 is the moment from when $PH_{t_0} \neq 0$. ■

In Proposition 1, λ_t is computed according to inner variable $\bar{p}_t$ which reveals the changing trend of p_t. The only shortcoming is the empirically defined constant f. Fortunately, the meaning of f is the number of waiting steps before making the decision. It is independent and has no relationship with any other variables, e.g., p_t and u_t. Therefore, we can set it to be a common value, e.g., 30, which is not too large to detect changes in time and not too small to collect sufficient evidence for making a decision.

The parameter δ in PH is a tolerance parameter that makes the PH test more robust when dealing with slowly varying distribution. If $\delta = 0$, the m_t in equation (1) may keep decreasing when p_t is decreasing or slowly increasing but lower than $\bar{p}_t$. Then the detection of change is impatiently reported before p_t slowly increasing above $\bar{p}_t$. However, δ should not be set to a large value, because a large δ will dominate the increasing of m_t and the change of p_t on basis of $\bar{p}_t$ will be neglected. The detection of changes will be then overleaped. δ is usually set to a very small value in range of $[10^{-3}\ 10^{-1}]$. In this paper we set $\delta = 10^{-2}$.

III. Experimental results on EGEE jobs

We validate our self-adaptive change detection approach in the framework of StrAP method used in [10]. StrAP combines a clustering method called Affinity Propagation (AP) with the change point detection test. The clustering method AP is used for summarizing the streaming data and change detection is used for catching the non-stationary distribution in the streaming data. In the original StrAP framework, there are several ways for detecting the changes. One way is called "MaxR" which detects the changes through specifying the maximum number of outliers referred to as "size of reservoir". The other ways are based on PH test with threshold λ specified by a fixed-value, or with threshold λ adapted by optimization approaches, e.g., e-greedy selection and Gaussian Process Regression [10].

In the validation of this paper, we will use the self-adaptive change detection approach in StrAP framework. The goal of the validation is to show that the self-adatpive change detection approach can achieve better clustering accuracy, comparing to the other ways in original StrAP method [10]. The cluster accuracy measures the model of StrAP in a fashion of supervised learning, where the labels of data items are known from the prior knowledge. In StrAP model, each data item is assigned to one cluster, and its label is expected to be the same as the label of the exemplar in its assigned cluster. The accuracy of clustering is defined as

$$\text{Clustering accuracy} = \frac{\sum_{i=1}^{K} |C_i^e|}{N} \tag{3}$$

where $|C_i^e|$ is the number of data items whose labels are the same as the exemplar's label in cluster C_i, K is the number of clusters, and N is the total number of data items. Higher accuracy means the better quality of the model in terms of summarizing the patterns of data steams and tracking the evolving distribution of data steams.

The data set we used for validation is the EGEE jobs. As described in [3], this data set includes 5,268,564 jobs from EGEE grid during 5 months (from 2006-01-01 to 2006-05-31). Each job is labeled by its final state, successfully finished (*good job*) or failed (*bad job*, including about 45 error types. The 5 million jobs include about 20 main error

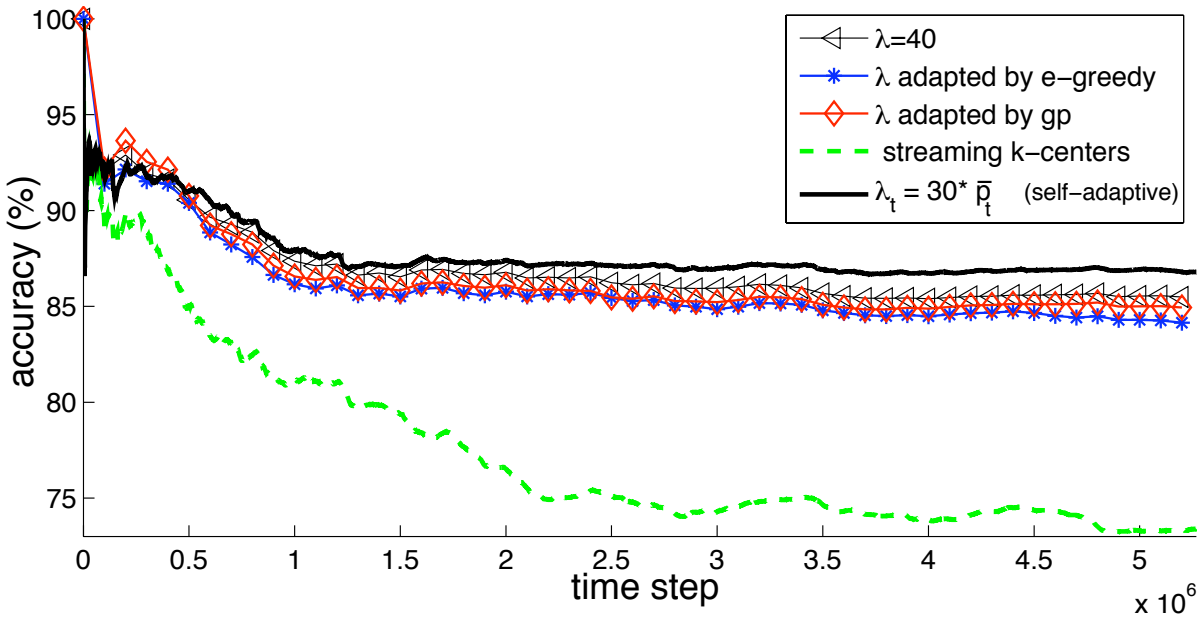

Fig. 2. Performance of StrAP on EGEE jobs: using the self-adaptive change detection approach $\lambda_t = 30 * \bar{p}_t$, λ adapted by e-greedy and Gaussian Process in [10], and λ fixed by a given value 40.

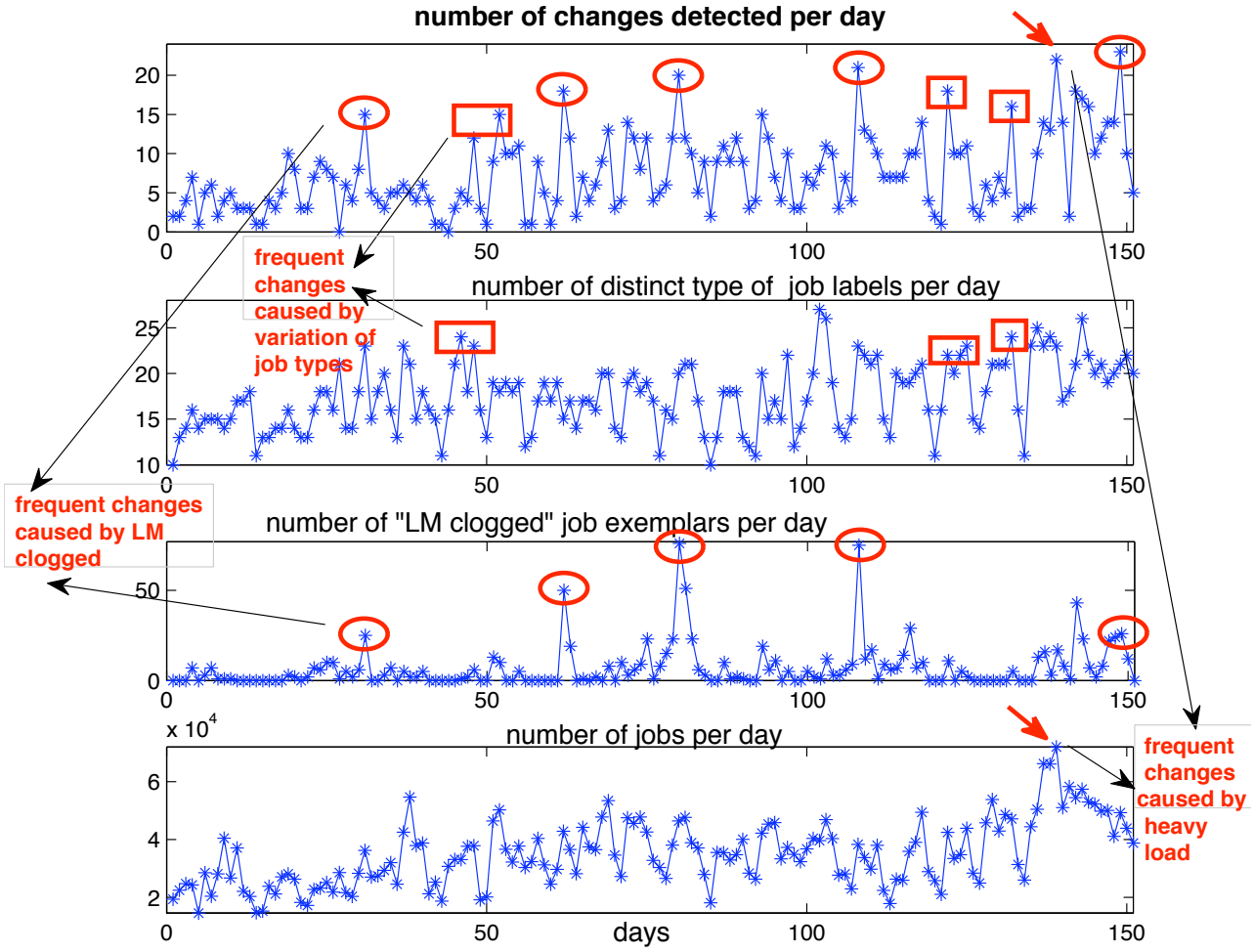

Fig. 3. Frequently detecting the changes because of the variation of job types (as claimed in the labels), or because of the heavy load per day, or because of the device fault (LogMonitor is clogged) which is not claimed in the labels but discovered by the change detection process and StrAP model.

types (more than 1,500 occurrences). Using StrAP method on these streaming jobs produces the visible summarized results to the administrators. The clustering accuracy is evaluated to guarantee that the compact description is correct.

To be clear, we need to explain why the lables of jobs are not used in the stream clustering process but why they are used for evaluation. As suggested by the experts, the labels of jobs might introduce some confusion, because they do not indicate the properties of the jobs regarding the failure reasons. Therefore, we do not use the labels in the clustering process, by contrast we aim to discover something not claimed in the label. However, since there is no reference interpretation, the labels could be the approximation of the classes of jobs although they are not precise enough. At least, the labels distinguish the good (successfully finished) jobs from the bad (failed) jobs.

In Fig. 2, we show the clustering accuracy of StrAP for clustering the EGEE job streams. We compare the performance of different change detection approaches, our proposed self-adaptive approach (real-time adapted threshold $\lambda_t = 30 * \bar{p}_t$), and the λ adapted by e-greedy and Gaussian Process in [10], and the λ fixed by a given value 40. Meanwhile, we use the *streaming k-centers* as the baseline for comparison, which exactly uses the same framework of StrAP, just replacing clustering method Affinity Propagation with k-centers.

From Fig. 2, it can be seen that our proposed self-adaptive change detection approach with real-time adapted threshold λ_t has higher accuracy than all the other approaches.

Fig. 3 shows the frequently detected changes on some special days, and the corresponding reasons causing the changes. The top figure in Fig. 3 shows the number of changes detected per day by our proposed method. The second figure in Fig. 3 is the number of distinct type of labels claimed in the logs of jobs. Since there is only 1 type of *good job* but 45 types of *bad jobs*, the number of distinct type of labels is indeed the number of distinct type of errors of failed jobs. The third one shows the number of job exemplars (clusters) which are the anomalies in StrAP model. After analyzing the time durations of these abnormal jobs spent on different services, we find that these anomalies are related a device fault called "LogMonitor is clogged", which was not claimed in the job labels. The bottom figure demonstrates the number of submitted jobs per day.

Through analyzing the number of changes detected per day in the top figure of Fig. 3, we see that changes are detected more than 15 times in around 10 different days. On the next day of the peak days, the number of changes is usually largely dropped. The decrease of the number of detected changes after peaks indicates that the StrAP model has caught the changes and was well-updated for handling the changed distribution.

An interesting thing is to investigate the reasons why changes are frequently detected in these days. Comparing the peak days in the top figure and the peak days in the other 3 figures in Fig. 3, we find that on 3 days the frequent changes are related to the large number of different types of errors of the jobs failed in one day (marked by the red squares). On another 5 days frequent changes are caused by the device fault "LogMonitor is clogged" (marked by the red circles). The last frequent change happened on day149 is caused by the heavy load of submitted jobs.

From the experimental results and analysis, we see that our proposed approach of change detection can firstly effectively discover the changes of the distribution in streaming data and make the StrAP model catch better the evolving distribution to achieve more accurate clustering results (Fig. 2). Secondly, the frequency of detected changes is related to the appearance of the unusual grid status, e.g. heavy load, large number of distinct errors and specific device fault (Fig. 3). It is important to note that the discovered fault of device is not claimed in the job labels, but firstly suspected by causing the frequent detection of changes and then confirmed by the StrAP model.

IV. Conclusion

In this paper, we proposed a self-adaptive change detection approach for detecting the changes in EGEE gird jobs with non-stationary distribution. This approach is based on the Page-Hinkley statistic test. The threshold λ for deciding the detection of changes is self-adapted instead of setting by a fixed value. Embedded in the streaming clustering process, StrAP, this self-adaptive change detection approach shows its flexibility and better performance on higher accuracy as reported in the experimental results. There are several perspectives opened by this approach. Firstly we will apply this self-adaptive change detection approach on more application fields, e.g. anomaly detection in grid computing system and in network traffic. Second, we will diagnose the different types of changes happened in the non-stationary distribution, such as *dissolution*, *coagulation* and *shift*.

References

[1] G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation," in *ICAC '06*, 2006, pp. 65–73.

[2] I. Rish, M. Brodie, and S. M. et al, "Adaptive diagnosis in distributed systems," *IEEE Transactions on Neural Networks*, vol. 16, pp. 1088–1109, 2005.

[3] X. Zhang, M. Sebag, and C. Germain-Renaud, "Multi-scale real-time grid monitoring with job stream mining," in *9th IEEE International Symposium on Cluster Computing and the Grid (CCGrid)*, 2009, pp. 420–427.

[4] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi, "An information-theoretic approach to detecting changes in multi-dimensional data streams," in *Proceedings of 38th Symposium on the Interface of Statistics, Computing Science, and Applications (Interface '06)*, 2006.

[5] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Statistical change detection for multi-dimensional data," in *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2007, pp. 667–676.

[6] C. C. Aggarwal, "A framework for diagnosing changes in evolving data streams," in *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, 2003, pp. 575–586.

[7] X. Zhang, C. Furtlehner, and M. Sebag, "Data streaming with affinity propagation," in *ECML/PKDD*, 2008, pp. 628–643.

[8] E. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, pp. 100–115, 1954.

[9] D. Hinkley, "Inference about the change-point from cumulative sum tests," *Biometrika*, vol. 58, pp. 509–523, 1971.

[10] X. Zhang, C. Furtlehner, J. Perez, C. Germain-Renaud, and M. Sebag, "Toward autonomic grids: Analyzing the job flow with affinity streaming," in *ACM SIGKDD*, 2009, pp. 987–995.
[11] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, pp. 461–464, 1978.